2

PATENT ABSTRACTS OF JAPAN

(11)Publication number : 11-338521
(43)Date of publication of application : 10.12.1999

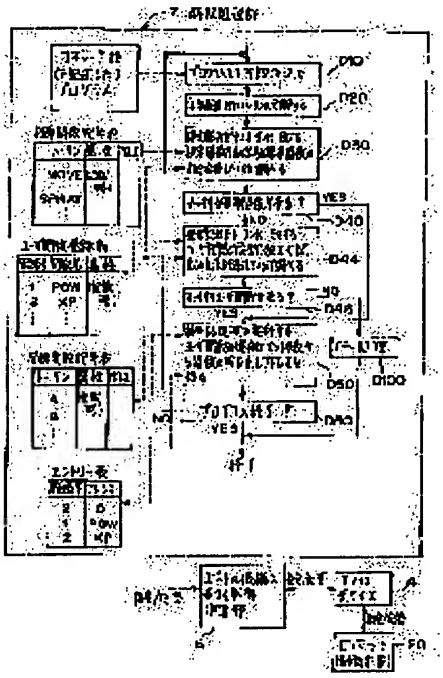
(51)Int.Cl. G05B 19/05
B25J 9/16
G06F 9/06
G06F 9/45

(21)Application number : 10-147267
(22)Date of filing : 28.05.1998

(71)Applicant : MITSUBISHI HEAVY IND LTD
(72)Inventor : ISHII KEI

(54) CONTROLLER

(57)Abstract:
PROBLEM TO BE SOLVED: To enable a user to make a controlled system perform necessary operation without remodeling the system itself and without decreasing the operation efficiency even when the controlled system performs complicated operation, in a controller, which is used suitably to control a general robot and a plant.
SOLUTION: The controller equipped with an interpreting process part 7 which performs a language interpreting and arithmetic process by using standard functions for a control program described in a robot language or graphic language and a control arithmetic part 6 which commands the controlled system 20 to operate according to command information obtained by the language interpreting and arithmetic process of the interpreting process part 7 is equipped with a user function registering function which allows a user to register user functions different from the standard functions, and the interpreting process part 7 performs the language interpreting and arithmetic process by using the user functions for a control program corresponding to the user functions.



LEGAL STATUS

[Date of request for examination]

[Date of sending the examiner's decision of rejection]

[Kind of final disposal of application other than the examiner's decision of rejection or application converted registration]

[Date of final disposal for application]

[Patent number]

[Date of registration]

[Number of appeal against examiner's decision of rejection]

(19)日本国特許庁（ＪＰ）　(12) **公開特許公報**（Ａ）　(11)特許出願公開番号

特開平11－338521

(43)公開日　平成11年(1999)12月10日

(51)Int.Cl.[6]		識別記号	FI		
Ｇ０５Ｂ	19/05		Ｇ０５Ｂ	19/05	Ａ
Ｂ２５Ｊ	9/16		Ｂ２５Ｊ	9/16	
Ｇ０６Ｆ	9/06	５３０	Ｇ０６Ｆ	9/06	５３０Ｃ
	9/45			9/44	３２０Ｃ

審査請求　未請求　請求項の数１　ＯＬ　（全 12 頁）

(21)出願番号　特願平10－147267

(22)出願日　平成10年(1998) 5月28日

(71)出願人　000006208
三菱重工業株式会社
東京都千代田区丸の内二丁目５番１号

(72)発明者　石井　圭
兵庫県高砂市荒井町新浜２丁目１番１号
三菱重工業株式会社高砂研究所内

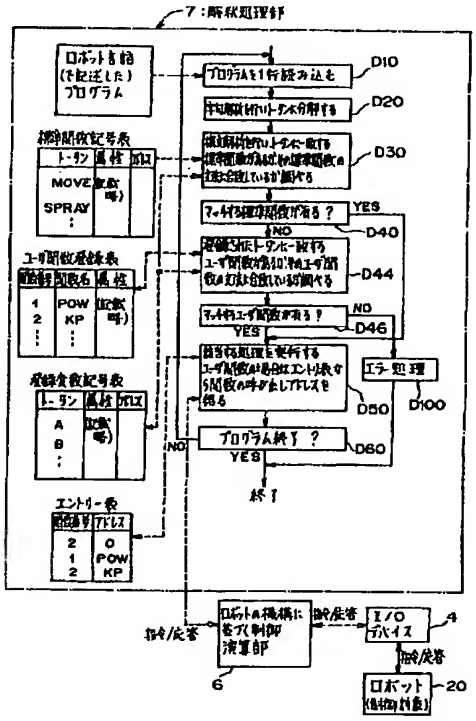
(74)代理人　弁理士　真田　有

(54)【発明の名称】　制御装置

(57)【要約】

【課題】　汎用ロボットやプラントの制御に用いて好適の制御装置において、システム自体の改造をすることなく、かつ、制御対象に複雑な作業を行なわせる場合にも作業効率を低下させることなく、ユーザが必要な動作を制御対象にさせることのできるようにする。

【解決手段】　ロボット言語又は図形言語により記述された制御プログラムに対して標準関数を用いて言語解釈及び演算処理を行なう解釈処理部７と、解釈処理部７による該言語解釈及び該演算処理によって得られた指令情報に従って制御対象２０に動作指令を行なう制御演算部６とをそなえた制御装置において、ユーザが該標準関数とは別のユーザ関数を登録するユーザ関数登録機能をそなえ、解釈処理部７は、該ユーザ関数に対応した該制御プログラムに対しては該ユーザ関数を用いて言語解釈及び演算処理を行なうように構成する。



【特許請求の範囲】
【請求項１】　ロボット言語又は図形言語により記述された制御プログラムに対して標準関数を用いて言語解釈及び演算処理を行なう解釈処理部と、該解釈処理部による該言語解釈及び該演算処理によって得られた指令情報に従って制御対象に動作指令を行なう制御演算部とをそなえた制御装置において、
ユーザが該標準関数とは別のユーザ関数を登録するユーザ関数登録機能をそなえ、
該解釈処理部は、該ユーザ関数に対応した該制御プログラムに対しては該ユーザ関数を用いて言語解釈及び演算処理を行なうことを特徴とする、制御装置。
【発明の詳細な説明】
【０００１】
【発明の属する技術分野】本発明は、ロボット言語により制御可能な汎用ロボットの制御、あるいは、図形言語によりプラント制御ロジックを記述して制御可能なプラントの制御に用いて好適の、制御装置に関する。
【０００２】
【従来の技術】従来より、ロボットの制御やプラントの制御を行なう制御装置が開発されている。このような従来の制御装置では、ユーザが容易に制御プログラムを記述して用途に応じた動作指令を制御対象に行なうことができるように、標準関数をそなえた基本ソフトウエアが用意されている。ユーザは、この標準関数を組み合わせて、目的に応じた制御プログラムを記述している。特に比較的に緻密な作業に使用されることの多いロボットでは、適宜に動作の修正又は変更を要求されることが多いため、ユーザが目的に応じてロボットに動作指令を行なわせることを容易にできる制御装置の要求は高い。
【０００３】ロボットとしては、例えば、塗装スプレーを行なう塗装ロボットがある。図７～図１０は従来の制御装置にかかる塗装ロボットを示す図である。図７は塗装ロボットの構成及び塗装ロボットの動作内容を示す図であり、図８はロボット言語（標準関数）による塗装ロボットの制御例を示す図であって、（ａ），（ｂ）はそれぞれ図７の動作を塗装ロボットに行なわせるためのロボット言語による制御プログラムを示す図であり、図９はロボット制御装置の内部構成を示す図であり、図１０はロボット言語により記述した制御プログラムの解釈処理のフローを説明する装置の要部構成図である。
【０００４】図７に示すように、塗装ロボット２０は、塗装スプレー２１と、塗装スプレー２１を移動させる移動機構２５と、塗装スプレー２１及び移動機構２５に動作指令を行なうロボット制御装置１０とから構成され、作業対象４０にスプレー塗装を行なう。ここで、移動機構２５は、軸２６ａ，２７ａ，２８ａ，２９ａと、これらの軸２６ａ～２９ａの軸間及び軸端に設けられた傾動関節２６，２７，２９及び回動関節２８，３０とをそなえている。傾動関節２６，２７，２９は、結合された軸を傾動できるようになっており、回動関節２８は、接続された軸をその軸心線回りに回動できるようになっている。回動関節３０は、塗装ロボット２０を支持しており、接続された軸をその軸心線回りに片方向に回転できるようになっている。
【０００５】これらの傾動関節２６，２７，２９，回動関節２８，３０には、ロボット制御装置１０からの動作指令で動作する図示されない関節駆動装置、例えば、モータ及び減速機がそれぞれ設置されている。また、傾動関節２６，２７，２９，回動関節２８，３０には、エンコーダがそれぞれ付設されており、これらのエンコーダにより検出した傾動関節２６，２７，２９，回動関節２８，３０の動作量をロボット制御装置１０にフィードバックしている。
【０００６】そして、ロボット制御装置１０の内部は、図９に示すように、ＣＰＵ１と、ＲＡＭ２と、ＲＯＭ３と、Ｉ／Ｏデバイス４と、バッテリバックアップ付きのＲＡＭ５と、それらを互いに繋げる伝送路９とから構成される。
【０００７】ここで、塗装ロボット２０を制御するための基本ソフトウェアはＲＯＭ３に格納されており、ＣＰＵ１はこの基本ソフトウェアをＲＡＭ２を使って実行する。制御プログラムは、図８に示すようなロボット言語により記述されており、電源を切っても内容が失われないメモリ領域、ここでは、バッテリバックアップ付きのＲＡＭ５に通常ロードされている。このバッテリバックアップ付きＲＡＭ５の代わりに、フラッシュＲＯＭなどを使うこともある。そして、ロボット制御装置１０はＩ／Ｏデバイス４を介して塗装ロボット２０とつながれている。
【０００８】塗装ロボット２０は上述のように構成されるので、図８（ａ），（ｂ）に示すようなロボット言語（標準関数）による制御プログラムを作成してロボット制御装置１０へ入力することで、塗装ロボット２０に作業対象物４０の所定の位置に塗装を行なわせることができる。この制御プログラムは、ロボット制御装置１０のＣＰＵ１で解釈処理された後に、Ｉ／Ｏデバイス４を介して、動作信号として塗装スプレー２１及び移動機構２５の上述の関節駆動装置に送られる。この動作信号に応じて、塗装スプレー２１は塗装物の噴射の開始又は停止の何れかを行ない、各関節駆動装置は互いに連動して塗装スプレー２１を所定の位置へと移動させる。
【０００９】ここで、図８（ａ）のロボット言語により記述された制御プログラムについて説明する。この記述内容は、塗装ロボット２０により、図７に示す作業対象物４０の、座標１，２間及び座標３，４間の範囲で塗装を行なわせるためのものである。ここで、座標１～４は何れも作業対象物４０の端点に位置する。以下に、図８（ａ）の制御プログラムを、図７と対応させながら、上から一行ずつ説明していく。

【００１０】「ＭＯＶＥ」は、塗装スプレー２１を、引数で指定した座標へ移動させるための関数である。ここでは、「ＭＯＶＥ　１」として引数に１をセットして、塗装スプレー２１を塗装の開始点で作業対象物４０の端点に位置する座標１へと移動させる（ステップＡ１）。次の「ＯＶＥＲＳＰＲＡＹ」は引数で指定した量をオーバー量に設定する関数である。ここで、オーバー量とは、塗装スプレー２１の、塗装物の噴射停止時における不安定な噴射状態による塗装むらを考慮して設定するもので、ＯＶＥＲＳＰＲＡＹ以降に記述されるＭＯＶＥについては、その引数よりもＯＶＥＲＳＰＲＡＹにより設定したオーバー量だけ離れた座標へと塗装スプレー２１を移動させる。さらに、ＯＶＥＲＳＰＲＡＹ以前にＭＯＶＥ指令により塗装スプレー２１を移動させている場合は、まず、このＭＯＶＥ指令による移動とは逆方向にオーバー量だけ塗装スプレー２１を引き戻す。

【００１１】ここでは、「ＯＶＥＲＳＰＲＡＹ　５０」としてオーバー量を５０mmに設定している。したがって、塗装スプレー２１は、直前のＭＯＶＥ（ステップＡ１）による移動に対して、ＯＶＥＲＳＰＲＡＹにより設定した量（オーバ量）だけ手前の図中Ａの位置へと引き戻される。（ステップＡ２）。

【００１２】塗装スプレー２１が図中Ａの位置へ移動した時点で、次の「ＳＰＲＡＹ　ＯＮ」により、塗装スプレー２１の塗装物の噴射を開始させる（ステップＡ３）。次に、「ＭＯＶＥ　２」として引数に２をセットして、塗装スプレー２１を、ＭＯＶＥ関数により、作業対象物４０の端点に位置する座標２へ移動させるように指令しているが、ステップＡ２のＯＶＥＲＳＰＲＡＹの設定により、ＭＯＶＥ関数により指定された位置（座標２）に対しＯＶＥＲＳＰＲＡＹにより設定した量（オーバ量）だけ先の位置（図中Ｂの位置）へと移動させる（ステップＡ４）。

【００１３】塗装スプレー２１が図中Ｂの位置へ移動した時点で、次の「ＳＰＲＡＹ　ＯＦＦ」により、塗装スプレー２１の塗装物の噴射を停止させる（ステップＡ５）。次の「ＭＯＶＥ　３」は、塗装スプレー２１を、ＭＯＶＥ関数により、座標３へ移動させるように指令しているが、ＭＯＶＥ関数により指定された位置（座標３）に対し、ＯＶＥＲＳＰＲＡＹで設定したオーバ量だけ手前の位置（図中Ｃの位置）へと移動させる（ステップＡ６）。

【００１４】塗装スプレー２１が図中Ｃの位置へ移動した時点で、次の「ＳＰＲＡＹ　ＯＮ」により、塗装スプレー２１の塗装物の噴射を開始させる（ステップＡ７）。次の「ＭＯＶＥ　４」は、塗装スプレー２１を、ＭＯＶＥ関数により、座標４へ移動させるように指令しているが、ＭＯＶＥ関数により指定された位置（座標４）に対し、ＯＶＥＲＳＰＲＡＹで設定したオーバ量だけ先の位置（図中Ｄの位置）へと移動させる（ステップＡ８）。

【００１５】塗装スプレー２１が図中Ｄの位置へ移動した時点で、次の「ＳＰＲＡＹ　ＯＦＦ」により、塗装スプレー２１の塗装物の噴射を停止させ（ステップＡ９）、塗装ロボット２０の作業が終了する。この図８（ａ）に示す制御プログラムにより、ロボット制御装置１０より動作指令を受けた塗装ロボット２０は、図７に示す作業対象物４０に対して、以下のように動作する。

【００１６】塗装スプレー２１は、まず、座標１の手前のある点から、作業対象物４０の端点に位置して塗装開始点である座標１の上方へと移動する（ステップＡ１）が、その後、図中のＡで示される位置まで後退して（ステップＡ２）、このＡの位置で塗装物の噴射を開始する（ステップＡ３）。そして、塗装スプレー２１は、塗装物を噴射しながら、移動機構２５により、このＡの位置から、座標１及び所定の座標２を通過して図中のＢで示される位置まで移動して（ステップＡ４）、塗装物の噴射を停止する（ステップＡ５）。塗装開始点（座標１）よりもの所定量（５０mm）手前の地点Ａから塗装物の噴射を開始したり、塗装物の噴射停止点（座標２）を所定量（５０mm）オーバーした地点Ｂまで塗装物を噴射させるのは、塗装スプレー２１が停止した状態で、塗装物の噴射開始または停止を行なうと、不安定な噴射状態による塗装むらが塗装物の噴射開始点や停止点（座標１，２）付近で生じるおそれがあるので、これを防止するためである。

【００１７】次に、塗装スプレー２１は、移動機構２５により図中Ｃの位置に移動し（ステップＡ６）て、このＣの位置で塗装物の噴射を開始し（ステップＡ７）、塗装物を噴射しながら移動機構２５により図中のＤで示される位置まで移動して（ステップＡ８）、塗装物の噴射を停止する（ステップＡ９）。

【００１８】塗装を行なう所定範囲（座標３，４間）よりも、手前の位置（座標Ｃ）から塗装物の噴射を開始して、先の位置（座標Ｄ）まで塗装物を噴射させるのは、上述と同様に、塗装物の噴射開始点及び噴射停止点（座標３，４）付近での不安定な噴射状態による塗装むらが生じるのを防ぐためである。以上で、塗装ロボット２０の作業は終了する。

【００１９】ここで、図８（ａ）の制御プログラムを、図８（ｂ）の制御プログラムのように、即ち、図８（ａ）の制御プログラムに対し、ステップＡ１のＭＯＶＥ　１とステップＡ２のＯＶＥＲＳＰＲＡＹ　５０の順序を逆にして記述することもある。この場合、図８（ａ）の制御プログラムのステップＡ１からステップＡ３にかかるスプレーロボット２０の動作は、前述の通り、図７において、塗装スプレー２１が、座標Ａを通りすぎて、塗装開始点である座標１の上方へと一旦移動した後に座標Ａの位置まで引き返して塗装物の噴射を開始するのに対し、図８（ｂ）の制御プログラムのステップ

Ｅ１からステップＥ３にかかるスプレーロボット２０の動作は、ＭＯＶＥ１以前にＯＶＥＲＳＰＲＡＹによりオーバ量が設定されているので、塗装スプレー２１は、座標１まで移動してからＡの地点まで引き返すといった動作なしに、直接に、塗装開始点である座標１よりもオーバ量だけ手前の位置である座標Ａに移動し、そして塗装物の噴射を開始する。

【００２０】図８（ｂ）の制御プログラムによる動作は、座標Ａでの塗装物の噴射を開始以降については、図８（ａ）の制御プログラムによる動作と同じであるので説明を省略する。

【００２１】次に、ロボット言語の解釈処理について説明する。ロボット言語の解釈処理は、基本ソフトウェアにより、例えば、図１０に示すように行なわれる。ここで、解釈処理部７は、言語解釈（ステップＢ１０～ステップＢ４０）と演算処理（ステップＢ５０）とを行ない、言語解釈及び演算処理後に指令を制御演算部６に出力する。制御演算部６は、この指令によってさらに指令値を計算してＩ／Ｏデバイス４を介して塗装ロボット２０へ動作指令を出力する。また、塗装ロボット２０からは、塗装ロボット２０の位置等のロボットの状態を示す信号がロボット制御装置１０にフィードバックされる。

【００２２】まず、基本ソフトウェアによって、図８（ａ），（ｂ）に示すような制御プログラムを１行づつ読み込み（ステップＢ１０）、そして、読み込んだプログラム行を字句解析を行なって最小単位の語句（トークン）に分解する（ステップＢ２０）。ここで、制御プログラムは、基本ソフトウエアで解釈処理しやすいように適当な中間言語にコンパイルされてロードされている場合もある。

【００２３】次いで、このトークンを、標準関数記号表及び登録変数記号表と比較して構文解析を行なう（ステップＢ３０）。ここで、標準関数記号表には、その属性（関数の引数の数及び引数の型等）を含んで標準関数が登録されている。さらに、この標準関数記号表には標準関数を実際に処理するためのアドレスも登録されており、速やかに標準関数の処理の実行が行なえるようになっている。

【００２４】また、登録変数記号表も、標準関数と同様の内容を持っている。つまり、登録変数記号表には、使用される変数が、変数の型等の属性を含んで登録され、さらに、変数の実際のアドレス等の情報も含まれており、変数の内容を速やかに検索できるようになっている。つまり、構文解析では、トークンと標準関数登録表及び登録変数記号表とを比較してトークンが標準関数と合致しているかを調査する。即ち、分解されたトークンが、上述の標準関数記号表に登録されているいずれかの標準関数と一致するとともにその標準関数の属性（標準関数の文法）に適っているか、さらに、標準関数の引数に変数が使われている場合は、その変数が登録変数記号表に登録されている変数のいずれかと一致するとともにその変数の属性に適っているかを調査する。

【００２５】以上で、構文解析が終了し、ステップＢ４０へと進む。ステップＢ４０では、トークンが標準関数に合致しているかの判定がおこなわれる。ここで、トークンが標準関数に合致していなければ、ステップＢ１００へと進みエラー処理が行なわれて、制御プログラムの解釈処理は未完のまま強制終了となるが、トークンが標準関数に合致していれば、ステップＢ５０へ進む。

【００２６】ステップＢ５０では、トークンと合致する標準関数に該当する処理を行なうように制御演算部６に指令を出してステップＢ６０へと進む。ここで、ステップＢ５０、即ち、解釈処理部７から、解釈処理の結果得られた動作命令や移動先の座標などの指令を受けると、制御演算部６は、さらに指令値を計算して、この指令値（動作指令）を、Ｉ／Ｏデバイス４経由でロボットの各アクチュエーター（塗装ロボット２０）に出力する。

【００２７】ステップＢ６０では、ステップＢ１０で読み込んだ制御プログラムの１行が制御プログラムの最終行かどうか、即ち、制御プログラムが終了かどうかの判定が行なわれ、制御プログラムが終了であれば解釈処理は終了するが、制御プログラムが終了でなければ、ステップＢ１０へと戻って、制御プログラムの次の行を読み込み、制御プログラムが終了するまで解釈処理（ステップＢ１０～ステップＢ６０）を繰り返す。

【００２８】以上で、基本ソフトウェアによるロボット言語の解釈処理のフローが終了する。このように、従来のロボット制御装置では、いくつかの簡単なロボット言語（標準関数）を組み合わせて制御プログラムを記述することでロボットに複雑な動作を行なわせることができるようになっている。ロボット言語としては、例えば、汎用的なプログラム言語であるＢＡＳＩＣに類似したものがＪＩＳ規格にも定められている。

【００２９】このようなロボット言語の他に、プラント制御装置では、図１１のようなファンクションブロック図などの図形形式の言語を使って制御プログラムを記述するものもある。図１１の制御プログラムは、プラント制御装置にアナログ入力されている、図１１中①で示す制御対象となるある状態量の制御目標値（Ｍｏｋｕｈｙｏ）と図１１中②で示すこの状態量の現状の制御状態（Ｓｅｉｇｙｏｒｙｏ）との差分を、図１１中③で示す減算関数により算出し、さらに、この差分を使って図１１中④で示す関数〔（Ｋ／（１＋Ｔｓ）〕により一次遅れを考慮した操作量を算出して、この操作量を、制御対象となる状態量の調節機器に指令値としてアナログ出力させる（図１１中⑤）ものである。

【００３０】

【発明が解決しようとする課題】しかしながら、このような従来の制御装置では、必要な動作を制御対象に行なわせるために必要な制御プログラム（必要制御プログラ

ム）を記述するのにユーザが使用できるのは、制御装置に標準で提供されているロボット言語（標準関数）に限られているため、以下の課題がある。

【００３１】制御プログラムにより例えば汎用ロボットのような制御対象に動作を行なわせるには、図１０に示すように、制御プログラムを構成する標準関数のそれぞれについて解釈処理が必要であり、このため、１つの制御プログラムを実行するのに要する解釈処理時間は制御プログラムを構成する標準関数の数に比例して増加する。したがって、制御対象に複雑な動作を行なわせる必要があって必要制御プログラムを構成する標準関数が多数になるほど、解釈処理時間が増大し、ロボットやプラント等の制御対象の作業効率の低下を招いてしまう。

【００３２】ここで、複数の標準関数をまとめて１つのマクロ関数としてユーザが定義可能な、マクロ機能を持つ制御装置もある。複数の標準関数からなる類似の動作を繰り返し制御対象に行なわせるときには、この複数の標準関数を１つのマクロ関数とすることによりユーザの制御プログラム入力作業の煩雑さを解消することができるが、マクロ関数を構成する複数の標準関数のそれぞれについて解釈処理を行なっているので、上述と同様で解釈処理時間を短くすることはできず、制御対象の作業効率の低下を緩和するには至らない。

【００３３】さらに、制御装置に標準で提供される標準関数では、必要制御プログラムを記述しきることが困難な場合もある。このような場合、ユーザは、制御装置のシステム自体を高価な費用を払ってメーカに改造してもらうか、また、何とか既存の標準関数を組み合わせて必要制御プログラムを記述できたとしても、演算速度の遅いものになったり、制御プログラムの作成（組み合わせ）に大幅に時間や労力を要してしまう。

【００３４】本発明はこのような課題に鑑み創案されたもので、ロボット言語又は図形言語により制御可能な制御装置において、システム自体の改造をすることなく、かつ、制御対象に複雑な作業を行なわせる場合にも作業効率を低下させることなく、ユーザが必要な動作を制御対象に行なわせることができるようにした、制御装置を提供することを目的とする。

【００３５】

【課題を解決するための手段】このため、請求項１記載の本発明の制御装置は、ロボット言語又は図形言語により記述された制御プログラムに対して標準関数を用いて言語解釈及び演算処理を行なう解釈処理部と、該解釈処理部による該言語解釈及び該演算処理によって得られた指令情報に従って制御対象に動作指令を行なう制御演算部とをそなえた制御装置において、ユーザが該標準関数と別のユーザ関数を登録するユーザ関数登録機能をそなえ、該解釈処理部は、該ユーザ関数に対応した該制御プログラムに対しては該ユーザ関数を用いて言語解釈及び演算処理を行なうことを特徴としている。

【００３６】

【発明の実施の形態】以下、図面により、本発明の実施形態について説明する。本実施形態では、本制御装置をロボットの制御に適用した場合を説明する。図１～図６は本発明の一実施形態にかかるロボット制御装置を示すもので、図１はロボット言語の解釈処理のフローを説明する本装置の要部構成図、図２はユーザが作成したユーザ関数を制御装置に登録するためのユーザ関数登録表の一例を示す図、図３はユーザが作成したユーザ関数の演算内容の記述例を示す図、図４はエントリテーブル作成のための記述例を示す図、図５はバッテリバックアップＲＡＭの領域割付例を示す図であって、（ａ）はユーザ関数登録表とエントリテーブルを別個に作成した場合の領域割付例を示す図、（ｂ）はユーザ関数登録表とエントリテーブルを１つにまとめて作成した場合の領域割付例を示す図、図６はユーザが作成したユーザ関数の使用例を示す図である。

【００３７】本実施形態のロボット制御装置（制御装置）のハード構成は、従来技術と同様になっており、図９に示すように、ＣＰＵ１と、ＲＡＭ２と、ＲＯＭ３と、Ｉ／Ｏデバイス４と、バッテリバックアップ付きのＲＡＭ５と、それらを互いに繋げる伝送路９とから構成される。そして、本ロボット制御装置には、従来の汎用の制御装置と同様に、ユーザが容易に制御プログラムを記述して用途に応じた動作をロボット（制御対象）に実行させることができるように専用のプログラム言語をそなえた基本ソフトウエアが用意されるとともに、さらに、この基本ソフトウエアにユーザ関数登録機能がそなえられており、ユーザがＣ言語等の一般プログラム言語を利用して、専用のプログラム言語が標準としてそなえる標準関数とは別のユーザ関数を登録できるようになっている。

【００３８】ここで、ユーザ関数登録機能では、ユーザ関数について、ユーザ関数名と、ユーザ関数の演算内容と、ユーザ関数の属性（関数の引数の数及び引数の型等）とを登録するとともに、ユーザ関数を実際に処理するためのアドレスの登録も行なうようになっている。本実施形態では、ユーザの入力が容易となるように、ユーザ関数登録機能を、関数登録と、関数の演算内容の記述と、エントリテーブルの作成との３つに分けて構成している。

【００３９】ここで、２つのユーザ関数ＰＯＷ（），ＫＰ（）の、Ｃ言語を利用したユーザ関数登録を例に挙げて、図２～図４によりユーザ登録機能を説明する。図２に示すように、ユーザ関数登録は、ユーザ関数に関数番号（図２中、最も左の欄）を割り付けるとともに、関数の属性として関数の戻り値の型，引数の個数，引数の型を登録するもので、基本ソフトウエアがそなえる表形式によるシステム（登録関数表）によりユーザが容易に入力を行なえるようにしている。図２は、関数番号１に関

数名ＰＯＷ（）を、関数番号２に関数名ＫＰ（）を登録した状態を示している。ここで、関数の戻り値の型及び引数の型は、後述のユーザ関数の演算内容の記述と一致するように、ｉｎｔ，ｆｌｏａｔといったＣ言語に準じた型を使用する。

【００４０】そして、ユーザ関数の演算内容はＣ言語により図３のように記述する。図３はＰＯＷ（）の演算内容についての記述であって、べき乗を計算させるものである。また、図示しないが、ＫＰ（）の演算内容もＣ言語により同様に記述する。これらの記述は、Ｃ言語のコンパイラにより、制御装置のＣＰＵで動作可能なように実行モジュール（機械語）に変換されるとともにユーザ関数を実際に処理するためのアドレス（エントリアドレス）も割り付けられる。

【００４１】ここで、ユーザ関数を実際に使うためには、コンパイラにより割り付けられた各ユーザ関数のエントリアドレスを、基本ソフトウエアが行なう解釈処理で速やかに分かるようにする必要がある。このため、各関数のエントリアドレスを登録するためのエントリテーブルを、例えば、図４に示すＣ言語の記述によって作成する。ここで、図４の図中に①で示される数はユーザ関数の登録数〔本実施例ではＰＯＷ（）及びＫＰ（）の２つ〕を表し、図４中に②，③で示される数は図２のユーザ関数登録表中の関数番号に一致させて、関数登録表とエントリテーブルとの関連を速やかに確認できるようになっている。以上でユーザ関数登録は終了する。

【００４２】ここで、上述のユーザ関数登録により作成した、ユーザ関数登録表，エントリアドレス，ユーザ関数の実行モジュールは、ロボット制御装置のバッテリバックアップ付きのＲＡＭ５（図９参照）に図５（ａ）に示すような割り付け構成でロードして、電源が切れてもロード内容が消えないようにしている。また、ユーザ関数登録表ロード領域及びエントリーテーブルのロード領域の先頭アドレスを固定して、ロボット言語解釈部から容易に参照可能なようにしている。

【００４３】図６は、上述のようにユーザが定義した関数であるＰＯＷ（）及びＫＰ（）の使用例を示している。ステップＣ１０で、ユーザ関数ＰＯＷ（）により変数Ｂの４乗を計算してこれを変数Ａに代入し、さらにステップＣ２０で、ユーザ関数ＫＰ（）により、ステップＣ１０で求めた変数Ａの値に２．５を乗じたものに、さらに変数Ｄに４．２を乗じたものを加えて、これを変数Ｃに代入している。ステップＣ３０では、標準関数であるＭＯＶＥの引数として、１０，２０，３０にステップＣ２０で求めた変数Ｃを加えたものの３つを取り、制御対象をこの３つの座標へ順次移動させるようにしている。

【００４４】本発明の一実施形態にかかる制御装置は、上述のように構成されるので、例えば、図１に示すように、解釈処理を行なうことができる。この解釈処理は、図１０により説明した従来技術の解釈処理に対し、ユーザ関数の解釈処理（ステップＤ４４～ステップＤ４６）を加えたものである。つまり、まず、基本ソフトウェアは制御プログラムを１行づつ読み込み（ステップＤ１０）、そして、読み込んだプログラム行の字句解析を行なって最小単位の語句（トークン）に分解する（ステップＤ２０）。

【００４５】次いで、このトークンを、標準関数記号表及び変数の登録変数記号表と比較して構文解析を行ないトークンが標準関数に合致しているかを調査する。即ち、分解されたトークンが上述の標準関数記号表に登録されているいずれかの標準関数と一致しているか及びその標準関数の属性（標準関数の文法）に適っているかを調査し、さらに、関数の引数に変数が使われている場合は、その変数が登録変数記号表に登録されている変数（登録変数）のいずれかと一致するとともにその登録変数の属性に適っているかを調査する（ステップＤ３０）。そしてステップＤ４０へと進む。

【００４６】ステップＤ４０では、トークンが標準関数に合致しているかの判定が行なわれれ、トークンが標準関数に合致していなければユーザ関数を対象とした構文解析（ステップＤ４４）へ進み、トークンが標準関数に合致していればステップＤ５０へ進んで標準関数に該当する処理を行なうように制御演算部６に指令を出す。ステップＤ４４では、ユーザ関数を対象とした構文解析を行ない、トークンとユーザ関数登録表及び登録変数記号表とを比較してトークンがユーザ関数と合致しているかを調査し、即ち、分解されたトークンが上述のユーザ関数登録表に登録されているいずれかのユーザ関数と一致しているか及びそのユーザ関数の属性（ユーザ関数の文法）に適っているかを調査し、さらに、ユーザ関数の引数に変数が使用されていれば、その変数が登録変数記号表に登録されている変数（登録変数）のいずれかと一致しているか及びその登録変数の属性に適っているかを調査する。そして、ステップＤ４６へ進む。

【００４７】ステップＤ４６では、トークンがユーザ関数に合致しているかどうかの判定が行なわれ、トークンがユーザ関数に合致していなければ、ステップＤ１００へと進みエラー処理が行なわれて制御プログラムの解釈処理は未完のまま強制終了となり、トークンがユーザ関数に合致していれば、関数番号の情報を伴ってステップＤ５０へ進む。

【００４８】ステップＤ５０では、トークンと合致する標準関数またはユーザ関数に該当する処理を行なうように制御演算部６に指令を出してステップＤ６０へと進む。ここで、トークンがユーザ関数に合致している場合はエントリ表からユーザ関数のアドレスの検索を速やかに行なうことができる、つまり、各ユーザ関数に割り当てられた関数番号を介して関数登録表とエントリー表とを関連付けているのでユーザ関数のアドレスの検索を容

易に行なうことができる。

【００４９】また、ステップＤ５０、即ち、解釈処理部７から、言語解釈の結果得られた動作命令や移動先の座標などの指令を受けると、制御演算部６は、さらに指令値を計算して、この指令値（動作指令）をＩ／Ｏデバイス経由でロボット２０の各アクチュエーターに出力する。ステップＤ６０では、ステップＤ１０で読み込んだ制御プログラムの１行が制御プログラムの最終行かどうか、即ち、制御プログラムが終了かどうかの判定が行なわれ、制御プログラムが終了であれば解釈処理は終了するが、制御プログラムが終了でなければ、ステップＤ１０へと戻って、制御プログラムの次の行を読み込み、制御プログラムが終了するまで解釈処理（ステップＤ１０～ステップＤ６０）を繰り返す。

【００５０】以上で、基本ソフトウェアによるロボット言語の解釈処理のフローが終了する。本発明の制御装置は、このように、ユーザが目的に応じて一般的なプログラム言語を利用してユーザがユーザ関数を設定できるので以下のような効果がある。ユーザは、目的の作業をロボットにさせるために必要な制御プログラム（必要制御プログラム）をＣ言語を利用して作成すると、制御装置のＣＰＵで動作可能なように実行モジュール（機械語）に変換されて１つのユーザ関数として登録されるので、処理速度の高速化を図れる。即ち、複数の標準関数を組み合わせたものを解釈処理する場合と、これに相当する１つのユーザ関数を解釈処理する場合とを較べると、前者では標準関数の数だけ図１０に示すような解釈処理フローの繰り返しが必要なのに対し、後者では図１に示すような解釈処理を１回行なうだけで良い。

【００５１】例えば、掛け算１個にかかる解釈処理時間を約３００μｓｅｃに対し、そのうちのＣＰＵでの演算時間は僅か約１μｓｅｃである。ここで、１０００個の掛け算が必要なロボットの座標計算を制御装置に行なわせると、従来の制御装置では１０００個の掛け算を１個ずつ解釈処理するので約３００ｍｓｅｃかかるが、本発明にかかる制御装置ではこの１０００個の掛け算を機械語に変換して１つのユーザ関数として登録できるので解釈処理は１回で済むため従来の制御装置に較べて、その解釈処理時間は２桁少なくなる。

【００５２】さらに、必要制御プログラムをロボット制御装置用に標準で提供される標準関数では記述できない場合でも、豊富な関数をそなえる一般的なプログラム言語であるＣ言語を利用してユーザ言語を作成することでこの必要制御プログラムを記述することが可能となる。したがって、ユーザは、ロボット制御装置のシステム自体を高価な費用を払ってメーカに改造してもらうことなく、あるいは、標準関数の組み合わせで苦心するという効率の悪い作業を強いられることなく、目的に応じた作業をロボットに行なわせることができる。

【００５３】なお、本発明は、上述の実施形態に限定されるものではない。例えば、本実施形態では、Ｃ言語によりユーザ関数登録を行なっているが、例えばＦＯＲＴＯＲＡＮのような他の一般的なプログラム言語を用いてもよい。また、本実施形態では、ユーザ登録機能において、ユーザ関数登録表とエントリテーブルとを別々に構成しているが、これに限定されるものではなく、ユーザ関数登録表とエントリテーブルとを１つにまとめた構成としてもよい。この場合、ロボット制御装置のバッテリバックアップ付きＲＡＭ内の領域構成は、例えば図５（ｂ）に示すようになる。

【００５４】さらに、本実施形態では、ユーザ関数登録表は基本ソフトウエアがそなえるシステムにより表形式で入力し、エントリテーブルはＣ言語により作成しているが、例えば、ユーザ関数登録表をＣ言語等の一般的なプログラム言語により作成したり、また、エントリテーブルがユーザ関数登録表と連動して自動的に作成されるシステムを基本ソフトウエアに設けてもよい。

【００５５】また、本実施形態では、本発明をロボットに適用しているが、本発明は、例えば、プラント等のロボット以外に適用してもよい。これに伴い、ロボット言語以外のプログラム言語、例えばファンクションブロック図（ＦＢＤ）等の図形言語に適用してもよい。この場合、ユーザは、図１１中で１個の四角枠で示される機能要素を一般的なプログラム言語を利用して作成，登録する。

【００５６】

【発明の効果】以上詳述したように、本発明の制御装置によれば、一般的なプログラム言語を利用して作成した、ユーザが目的の作業をロボットにさせるために必要な制御プログラム（必要制御プログラム）を、制御装置で動作可能なように実行モジュールに変換して１つのユーザ関数として登録することができるので、処理速度の高速化を図ることができる。さらに、必要制御プログラムをロボット制御装置用に標準で提供される標準関数では記述できない場合でも、豊富な関数をそなえる一般的なプログラム言語を利用して作成したユーザ関数によりこの必要制御プログラムを記述することが可能となる。

【００５７】したがって、ユーザは、制御装置のシステム自体を高価な費用を払ってメーカに改造してもらうことなく、あるいは、標準関数の組み合わせで苦心するという効率の悪い作業を強いられることなく、目的に応じた作業をロボットに行なわせることができる。

【図面の簡単な説明】

【図１】本発明の一実施形態にかかるロボット言語の解釈処理のフローを説明する本装置の要部構成図である。

【図２】本発明の一実施形態にかかるユーザ関数登録表の一例を示す図である。

【図３】本発明の一実施形態にかかるユーザ関数の演算内容の記述例を示す図である。

【図４】本発明の一実施形態にかかるエントリテーブル

作成のための記述例を示す図である。

【図５】本発明の一実施形態にかかるバッテリバックアップＲＡＭの領域割付例を示す図であって、（ａ）はユーザ関数登録表とエントリテーブルを別個に作成した場合の領域割付例を示す図、（ｂ）はユーザ関数登録表とエントリテーブルを１つにまとめて作成した場合の領域割付例を示す図である。

【図６】本発明の一実施形態にかかるユーザ関数の使用例を示す図である。

【図７】従来の制御装置にかかる塗装ロボットの構成及び塗装ロボットの動作内容を示す図である。

【図８】従来の制御装置にかかるロボット言語（標準関数）による塗装ロボットの制御例を示す図であって、（ａ），（ｂ）はそれぞれ図７の動作を塗装ロボットに行なわせるためのロボット言語による制御プログラムを示す図である。

【図９】従来の制御装置にかかるロボット制御装置の内部構成を示す図である。

【図１０】従来の制御装置にかかるロボット言語により記述した塗装ロボットの制御プログラムの解釈処理のフローを説明する装置の要部構成図である。

【図１１】従来の制御装置にかかるプラント制御装置で用いられるファンクションブロック図（ＦＢＤ）による制御プログラムの記述例を示す図である。

【符号の説明】

１　ＣＰＵ
２　ＲＡＭ
３　ＲＯＭ
４　Ｉ／Ｏデバイス
５　バッテリバックアップ付きのＲＡＭ
６　制御演算部
７　解釈処理部
９　伝送路
１０　ロボット制御装置（制御装置）
２０　塗装ロボット，ロボット（制御対象）
２１　塗装スプレー
２５　移動機構
２６，２７，２９　傾動関節
２６ａ，２７ａ，２８ａ，２９ａ　軸
２８，３０　回動関節
４０　作業対象物

【図２】

関数番号	関数名	戻り値の型	引数の個数	引数の型
1	POW	float	2	float, int
2	KP	float	4	float, float, float, float
3				

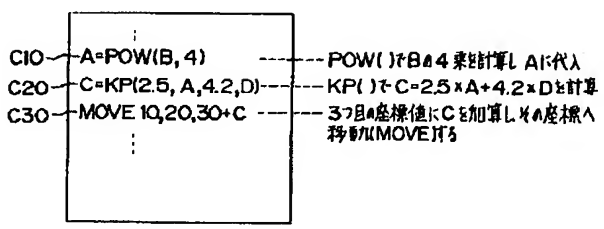
【図３】

```
/*POW()の演算記述例*/
float POW(a, b)
float a;
int   b;
{
      float c;
      int   i;
      for(i=0;c=1;i<b;i++){
            c=c*a;
      }
      return  c;
}
```

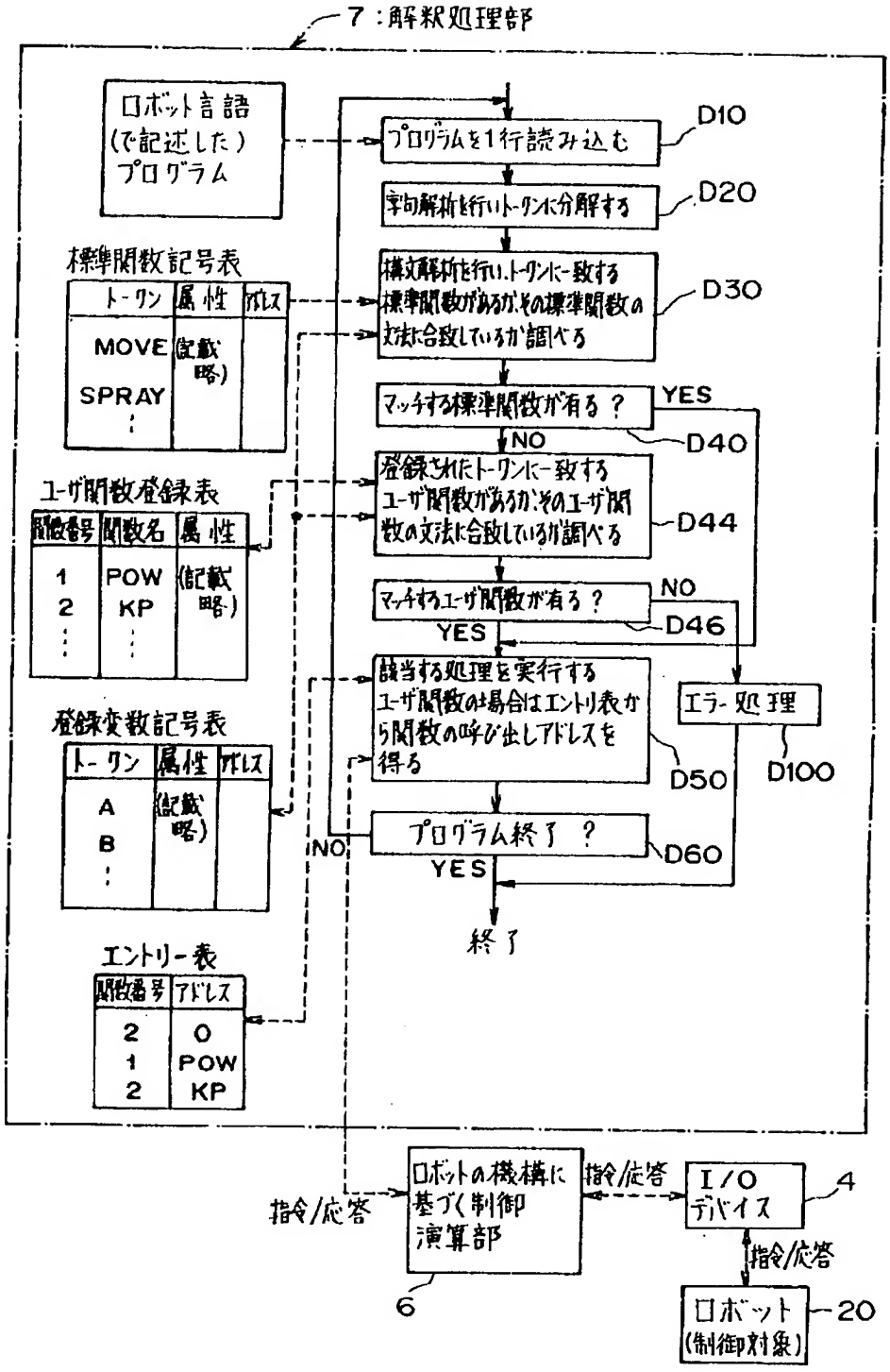
【図４】

```
/*エントリーテーブル記述例*/
extern float POW ();
extern float  KP();
struct ufunc_t{
      unsigned int no;
      float (*func)();
};
struct ufunc_t funtable[3]{
①       2,      0,
②       1,      POW,
③       2,      KP
};
```

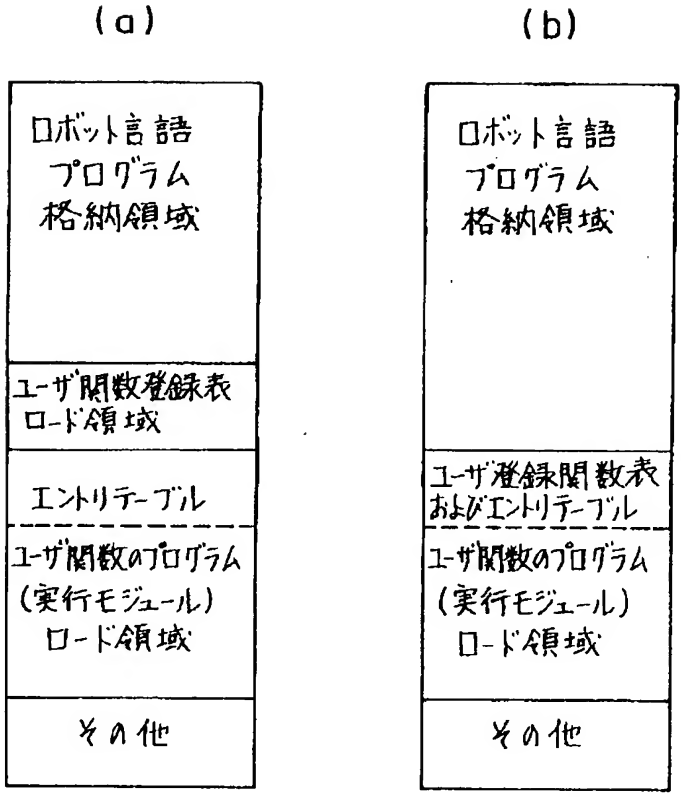
【図６】



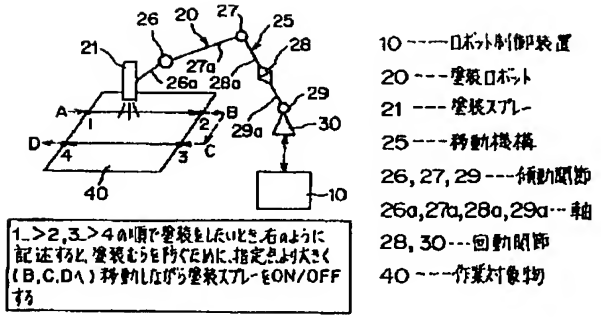
【図１】


7：解釈処理部
ロボット言語（で記述した）プログラム
プログラムを1行読み込む
D10
字句解析を行いトークンに分解する
D20
構文解析を行いトークンに一致する標準関数があるか、その標準関数の文法に合致しているか調べる
D30
マッチする標準関数が有る？
YES
NO
D40
登録されたトークンに一致するユーザ関数があるか、そのユーザ関数の文法に合致しているか調べる
D44
マッチするユーザ関数が有る？
NO
YES
D46
該当する処理を実行するユーザ関数の場合はエントリ表から関数の呼び出しアドレスを得る
D50
エラー処理
D100
プログラム終了？
NO
YES
D60
終了
標準関数記号表
トークン
属性
アドレス
MOVE
SPRAY
（記載略）
ユーザ関数登録表
関数番号
関数名
属性
1
2
POW
KP
（記載略）
登録変数記号表
トークン
属性
アドレス
A
B
（記載略）
エントリー表
関数番号
アドレス
2
1
2
0
POW
KP
指令/応答
ロボットの機構に基づく制御演算部
6
指令/応答
I/O デバイス
4
指令/応答
ロボット（制御対象）
20

【図５】



【図７】



【図８】

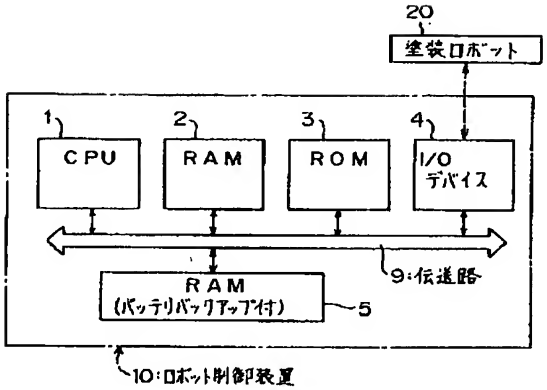
(a)

	<ロボット言語プログラム>	<説明>
A1	MOVE 1	座標1へ移動
A2	OVERSPRAY 50	オーバー量を50mmに設定
A3	SPRAY ON	塗装スプレーON
A4	MOVE 2	座標2の先のBへ移動
A5	SPRAY OFF	塗装スプレーOFF
A6	MOVE 3	座標3の手前Cへ移動
A7	SPRAY ON	塗装スプレーON
A8	MOVE 4	座標4の先のDへ移動
A9	SPRAY OFF	塗装スプレーOFF

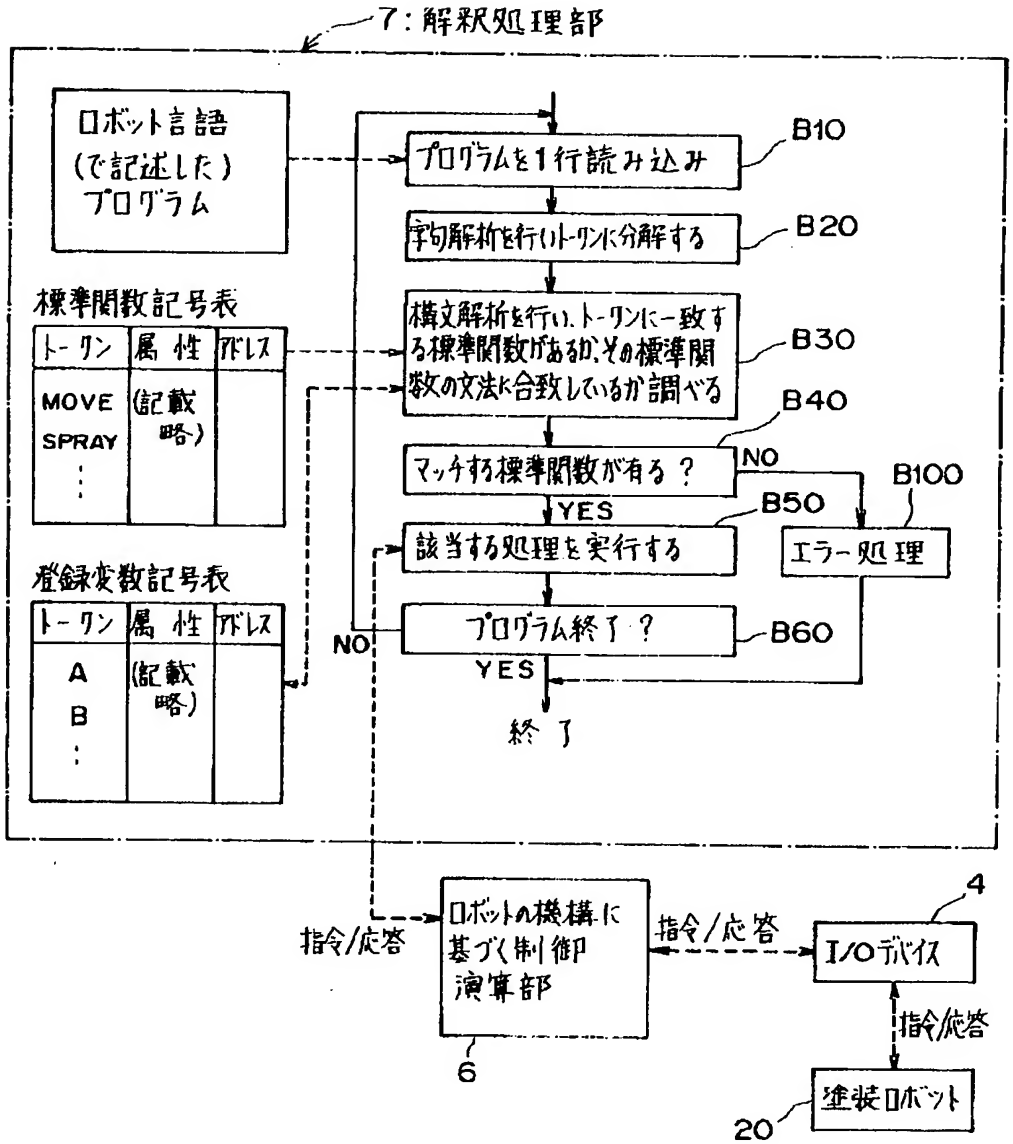
(b)

	<ロボット言語プログラム>	<説明>
E1	OVERSPRAY 50	オーバー量を50mmに設定
E2	MOVE 1	座標1の手前のAへ移動
E3	SPRAY ON	塗装スプレーON
E4	MOVE 2	座標2の先のBへ移動
E5	SPRAY OFF	塗装スプレーOFF
E6	MOVE 3	座標3の手前Cへ移動
E7	SPRAY ON	塗装スプレーON
E8	MOVE 4	座標4の先のDへ移動
E9	SPRAY OFF	塗装スプレーOFF

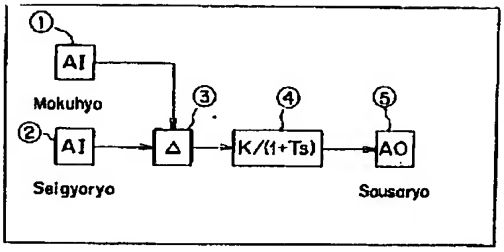
【図９】



【図１０】


7：解釈処理部
ロボット言語（で記述した）プログラム
プログラムを1行読み込み
B10
字句解析を行いトークンに分解する
B20
構文解析を行い、トークンに一致する標準関数があるか、その標準関数の文法に合致しているか調べる
B30
マッチする標準関数が有る？
B40
NO
YES
B100
エラー処理
該当する処理を実行する
B50
プログラム終了？
B60
NO
YES
終了
標準関数記号表
トークン
属性
アドレス
MOVE
SPRAY
（記載略）
登録変数記号表
トークン
属性
アドレス
A
B
（記載略）
指令/応答
ロボットの機構に基づく制御演算部
6
指令/応答
4
I/Oデバイス
指令/応答
塗装ロボット
20

【図１１】



AI　アナログ入力

AO　アナログ出力

△　減算

K/(1+Ts)　一次遅れ